

JP8305609

Publication Title:

PROGRAM TEST METHOD AND DEVICE

Abstract:

Abstract of JP 8305609

(A) Translate this text PURPOSE: To automatically carry out a large number of test programs and also to automatically decide the presence or absence of faults with no manual operation required by registering a test script for decision of faults in response to a test object program, carrying out this program, and also performing automatically the checking operations based on the test script. CONSTITUTION: A tester 30 inputs the execution contents of a test via a test script input means 10f to produce a test script for decision of faults. At the same time, the tester 30 selects a test mode by means of a mode switch/ execution means 10h. Then a test automation control means 10b gives a request to a test script management means 10a to acquire the information on the test script. Thus the means 10a sends the information on the test script to an automation control means 10c for a test object program.

Courtesy of <http://v3.espacenet.com>

特開平8-305609

(43)公開日 平成8年(1996)11月22日

(51)Int.Cl.⁶
G 0 6 F 11/28識別記号
3 4 0府内整理番号
7313-5BF I
G 0 6 F 11/28技術表示箇所
3 4 0 A

審査請求 未請求 請求項の数6 O.L (全18頁)

(21)出願番号 特願平7-104984

(22)出願日 平成7年(1995)4月28日

(71)出願人 000000295
沖電気工業株式会社
東京都港区虎ノ門1丁目7番12号(72)発明者 大江 哲男
東京都港区虎ノ門1丁目7番12号 沖電気
工業株式会社内

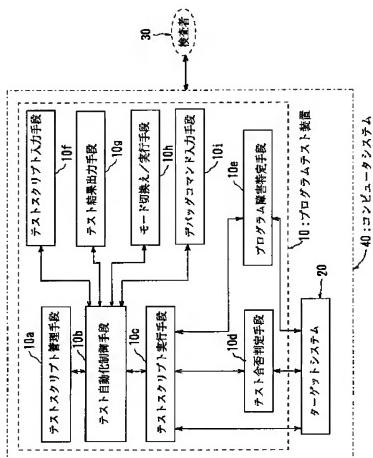
(74)代理人 弁理士 大垣 孝

(54)【発明の名称】 プログラムのテスト方法および装置

(57)【要約】

【目的】 プログラム中の障害の有無の判定、その障害の特定を、高いスキルを必要とすることなく自動的に行なえるプログラムのテスト方法を提供する。

【構成】 テスト対象のプログラムに障害があるか否かの判定に好適な障害有無判定用テストスクリプトをコンピュータの記憶部に登録する。テスト対象のプログラムをテスト実行すると共にこのテストスクリプトに基づくチェックを自動的に行ないその結果から障害の有無を判定する。上記方法で障害を含むと判定されたプログラムに対し次の処理をする。障害有無判定用テストスクリプトを、これによるテスト結果に基づき、障害特定に好適な障害特定用テストスクリプトに変更する。テスト対象のプログラムをテスト実行すると共にこの障害特定用テストスクリプトに基づくチェックを自動的に行ないその結果から障害を特定する。



【特許請求の範囲】

【請求項1】 テストスクリプトを登録するための書き換え可能なファイルを記憶部に設けておき、該ファイルに、テスト対象のプログラムに障害があるか否かの判定に好適な、該プログラムにおける所定ステップでの実行結果を出力させる旨および該実行結果に対するテスト項目を少なくとも含むテストスクリプトを、予め登録し、その後、

前記テスト対象のプログラムを実行すると共に、前記テストスクリプトに従い前記所定ステップでの実行結果のダンプおよび前記テスト項目によるテストを行なわせ、該行わせたテスト結果により前記テスト対象のプログラムに障害があるか否かを判定することを特徴とするプログラムのテスト方法。

【請求項2】 請求項1に記載のプログラムのテスト方法において、障害特定用テストスクリプトを登録するための書き換え可能なファイルを、前記記憶部に設けておき、請求項1に記載の処理によって障害があると判定された場合に、該ファイルに、請求項1の処理によって得られたテスト結果から見て当該障害の特定に好適な、前記テスト対象のプログラムを実行した際ににおける所定ステップでの実行結果を出力する旨および該実行結果に対するテスト項目を少なくとも含む障害特定用テストスクリプトを予め登録し、その後、

前記テスト対象のプログラムの全部または一部を再度実行すると共に、前記障害特定用テストスクリプトに従い前記好適なステップでの実行結果のダンプおよび前記好適なテスト項目によるテストを行なわせ、

該行なわせたテスト結果によりテスト対象のプログラムにおける障害箇所の特定を行なうことを特徴とするプログラムのテスト方法。

【請求項3】 請求項2に記載のプログラムのテスト方法において、前記障害特定用テストスクリプトは、テスト対象のプログラム中に障害があるか否かを判定するため登録した請求項1に記載の前記テストスクリプトを変更することにより作成することを特徴とするプログラムのテスト方法。

【請求項4】 プログラムに障害があるか否かをテストするための装置であって、テストスクリプトを記憶するための記憶手段を有したテストスクリプト管理手段と、

前記記憶手段に、テスト対象のプログラムに障害があるか否かの判定に好適な、該プログラムにおける所定ステップでの実行結果を出力させる旨および該実行結果に対するテスト項目を少なくとも含むテストスクリプトを、入力するための手段と、

前記テスト対象のプログラムを実行すると共に前記テストスクリプトに従い前記所定ステップでの実行結果のダ

ンプおよび前記テスト項目を実行する手段と、前記テスト項目を実行した際のテスト結果を出力する手段とを具えたことを特徴とするプログラムテスト装置。

【請求項5】 請求項4に記載のプログラムテスト装置において、障害の有無をテストするモードか障害を特定するモードかを選択する手段と、

障害特定用テストスクリプトを記憶するための手段を有したテストスクリプト管理手段と、前記障害を特定するモードが選択された場合に動作し、請求項4に記載のテスト結果から見て当該障害の特定に好適な、前記テスト対象のプログラムを実行した際ににおける所定ステップでの実行結果を出力する旨および該実行結果に対するテスト項目を少なくとも含む障害特定用テストスクリプトを入力するための手段と、前記テスト対象のプログラムの全部または一部を再度実行すると共に前記障害特定用テストスクリプトに従い前記好適なステップでの実行結果のダンプおよび前記テスト項目を実行する手段と、前記障害特定に好適なテスト項目を実行した際のテスト結果を出力する手段とをさらに具えたことを特徴とするプログラムテスト装置。

【請求項6】 請求項5に記載のプログラムテスト装置において、前記障害特定用テストスクリプトを入力するための手段は、請求項4に記載の障害の有無の判定に好適な前記テストスクリプトを変更することで当該入力を行なう手段であることを特徴とするプログラムテスト装置。

【発明の詳細な説明】**【0001】**

【産業上の利用分野】この発明は、プログラムに障害があるか否かを自動的にテストする方法およびその実施に好適な装置、また、プログラムに障害があった際には障害箇所の特定を容易にできる方法およびその装置に関するものである。

【0002】

【従来の技術】プログラムに障害があるか否かをテストするための従来方法の一つとして、テスト対象のプログラム中に予めいくつかのブレークポイントを設定し、これらブレークポイント毎に情報をダンプしてこの結果からデータが正しく処理されているかを人手によって確認する方法があった。また、プログラムのデバッグを目的としプログラムの実行履歴を蓄積し障害箇所を特定する方法として、例えば特開平4-162151号公報に開示されているように、障害解析関数をプログラムが作成しこれをデバッグ制御プログラムに組み込むことによって人手を介すことなく障害箇所を命令単位で特定する方法があった。

【0003】

【発明が解決しようとする課題】しかしながら、上述し

た従来方法のうちの、テスト対象のプログラム中に予めいくつかのブレークポイントを設定する等する前者の方法は、各ブレークポイント毎に情報をダンプして正しくデータが処理されているかの確認作業を人手によって行なうことによる効率の悪さ及び確認ミスの発生という問題があった。

【0004】また、障害解析関数をデバッグ制御プログラムに組み込み、障害箇所を特定するという従来方法はあくまでデバッグを目的とするものであり障害の有無の判定を自動的に行なうものではない。従って、障害の有無のテストを人手を介さず自動的に行なうことは不可能であるという問題があった。さらに、障害を特定するための障害解析関数の作成には、この関数が組み込まれるデバッグ制御プログラムを熟知して障害解析関数を作成する必要があるので、高いスキルが要求されると共に時間がかかるという問題があった。

【0005】

【課題を解決するための手段】そこで、この出願の第一発明によれば、テスト対象のプログラム中に障害があるか否かを判定するに当たり、テストスクリプトを登録するための書き換え可能なファイルを記憶部に設けておく。そして、このファイルに、テスト対象のプログラムに障害があるか否かの判定に好適な、該プログラムにおける所定ステップでの実行結果を出力させる旨および該実行結果に対するテスト項目を少なくとも含むテストスクリプト（以下、「障害有無判定用テストスクリプト」ともいいう。）を、予め登録しておく。そして、その後、前記テスト対象のプログラムを実行すると共に、前記障害有無判定用テストスクリプトに従い前記所定ステップでの実行結果のダンプおよび前記テスト項目によるテストを行なわせ、該行わせたテスト結果により前記テスト対象のプログラムに障害があるか否かを判定する。ただし、この第一発明および後の第三発明において、所定ステップは、テスト内容に応じ1箇所または複数箇所と出来、また、テスト項目はテスト内容に応じ1項目または複数項目とできる。

【0006】また、この出願の第二発明によれば、テスト対象のプログラム中に生じている障害を特定するに当たり、障害特定用テストスクリプトを登録するための書き換え可能なファイルを、前記記憶部に設けておく。そして、上記第一発明の方法に係る処理によって障害があると判定された場合に、該ファイルに、上記第一発明の方法に係る処理によって得られたテスト結果から見て当該障害の特定に好適な、前記テスト対象のプログラムを実行した際ににおける所定ステップでの実行結果を出力する旨および該実行結果に対するテスト項目を少なくとも含む障害特定用テストスクリプトを予め登録しておく。そして、その後、前記テスト対象のプログラムの全部または一部を再度実行すると共に、前記障害特定用テストスクリプトに従い前記好適なステップでの実行結果のダ

ンプおよび前記好適なテスト項目によるテストを行なわせ、該行わせたテスト結果によりテスト対象のプログラムにおける障害箇所の特定を行なう。ただし、この第二発明および後の第四発明において、好適なステップはテスト内容に応じ1箇所または複数箇所と出来、また、好適なテスト項目はテスト内容に応じ1項目または複数項目とできる。また全部または一部を再実行させる意味は、障害特定がテスト対象のプログラムの一部再実行で済む場合を考慮した意味である。

【0007】なお、この第二発明の実施に当たり、前記障害特定用テストスクリプトは、テスト対象のプログラム中に障害があるか否かを判定するためファイルに記憶してある障害有無判定用テストスクリプトを変更することにより作成するのが好適である。

【0008】また、この出願の第三発明によれば、プログラムに障害があるか否かをテストするための装置において、以下の(a)～(d)の各手段を具えたことを特徴とする。

【0009】(a). テストスクリプトを記憶するための記憶手段を有したテストスクリプト管理手段。

【0010】(b). 前記記憶手段に、テスト対象のプログラムに障害があるか否かの判定に好適な、該プログラムにおける所定ステップでの実行結果を出力させる旨および該実行結果に対するテスト項目を少なくとも含むテストスクリプト（すなわち、障害有無判定用テストスクリプト）を入力するための手段。

【0011】(c). 前記テスト対象のプログラムを実行すると共に前記テストスクリプトに従い前記所定ステップでの実行結果のダンプおよび前記テスト項目を実行する手段。

【0012】(d). 前記テスト項目を実行した際のテスト結果を出力する手段。

【0013】また、この出願の第四発明によれば、プログラムに障害があるか否かをテストするための装置において、上記(a)～(d)の各手段に加え、以下の(1)～(5)の各手段をさらに具えたことを特徴とする。

【0014】(1). 障害の有無をテストするモードか障害を特定するモードかを選択する手段。

【0015】(2). 障害特定用テストスクリプトを記憶するための手段を有したテストスクリプト管理手段。

【0016】(3). 前記障害を特定するモードが選択された場合に動作し、上記第三発明に係る方法によって得られたテスト結果から見て当該障害の特定に好適な、前記テスト対象のプログラムを実行した際ににおける所定ステップでの実行結果を出力する旨および該実行結果に対するテスト項目を少なくとも含む障害特定用テストスクリプトを入力するための手段。

【0017】(4). 前記テスト対象のプログラムの全部または一部を再度実行すると共に前記障害特定用テストスクリプトに従い前記好適なステップでの実行結果のダン

アおよび前記テスト項目を実行する手段。

【0018】(5)、前記障害特定に好適なテスト項目を実行した際のテスト結果を出力する手段。

【0019】

【作用】この出願の第一発明の構成によれば、テスト対象のプログラムを試験のために実行させた際に、このテスト対象のプログラムに適した任意のテスト内容を持つ障害有無判定用テストスクリプトに基づくチェックが自動的になされ、その結果が出力される。このため、例えばオペレータはこのチェック結果をみると、このテスト対象のプログラム中に障害があるか否かを判定できる。

【0020】また、この出願の第二発明の構成によれば、障害が生じているテスト対象のプログラムを再度実行させた際に、このテスト対象のプログラムにおける障害を特定するために適した任意のテスト内容を持つ障害特定用テストスクリプトに基づくチェックが自動的になされ、その結果が出力される。このため、例えばオペレータはこのチェック結果をみると、このテスト対象のプログラム中における障害の特定ができる。またさらに、特定をより進める為の更なる障害特定用スクリプトを作成する必要がある場合はそのデータが得られる。またここで用いる障害特定用スクリプトは、第一発明の方で得られたテスト結果に基づいて作成される。また上記更なる障害特定用スクリプトは、前回の障害特定用テストスクリプトによるテストで得られた結果に基づいて作成される。このため、障害特定用スクリプト（更なる障害特定用スクリプトの場合も含む。以下同様。）に含まれるべきチェック内容（ダンプを取りべきステップやテストすべき項目など）を、的確なものとし易いので、熟練をそれほど必要とすることなく障害特定用テストスクリプトを作成できる。

【0021】また、この出願の第三発明のプログラムテスト装置によれば、第一発明の実施を容易とする。また、この出願の第四発明のプログラムテスト装置によれば、第二発明の実施を容易とする。

【0022】

【実施例】以下、図面を参照してこの出願のプログラムのテスト方法およびプログラムテスト装置の実施例について説明する。ただし、いずれの図もこれらの発明を理解出来る程度に概略的に示してある。

【0023】1. プログラムテスト装置の説明

図1は、第一発明および第二発明の実施に好適なプログラムテスト装置10（すなわち第三および第四発明の実施例の装置）の説明図である。ただし、図1では、プログラムテスト装置10と、テスト対象のプログラムを実行する手段としてのターゲットシステム20と、検査者例ええばオペレータ30との関係も併せて示してある。ここで、このプログラムテスト装置10はコンピュータシステム40の一部で構成してある。また、このコンピュ

ータシステム40は、その一部で、組み込みシステムをデバッグするデバッガ（図示せず）をも構成しており、プログラムテスト装置10はこのデバッガが有する機能とリンクすることが出来る。もちろん、プログラムテスト装置10をターゲットシステム20を含む概念とする場合があつても良いし、デバッガと切り離した構成のものとしても良く、汎用的な構成にすることができる。

【0024】このプログラムテスト装置10は、テストスクリプト管理手段10aと、テスト自動化制御手段10bと、テストスクリプト実行手段10cと、テスト合否判定手段10dと、プログラム障害特定手段10eとを具える。さらにこのプログラムテスト装置10は、検査者30との間のインターフェース手段として、テストスクリプト入力手段10fと、テスト結果出力手段10gと、モード切換え／実行手段10hと、デバッグコマンド入力手段10iとを具える。これらインターフェース手段10f～10iは、それぞれ、テスト自動化制御手段10bと接続してある。以下、これら構成成分10a～10iについて詳細に説明する。

【0025】先ず、テストスクリプト管理手段10aは、テストスクリプト全体を管理するものであり、テストの実施内容が記述された1または複数のテストスクリプトの登録、削除、さらに、登録されたテストスクリプトの内容（これをテストスクリプト・ファイル情報という）の入出力や、テストの順番、テストの合否判定情報等から成るテストスクリプト管理情報の入出力等をするものである。具体的には、この実施例の場合、テストスクリプト管理手段10aは、以下に図2を参照して説明するような動作をするものとしてある。すなわち、テストスクリプトの登録か、テストスクリプトの削除か、テストスクリプト・ファイル情報の要求か、テストスクリプト管理情報の要求か、テストの終了かの各動作中から、検査者の希望するいずれかの動作を選択する（図2のステップS1～S5）。そして、それぞれの動作に応じ、テストスクリプトの登録およびその結果をテスト自動化制御手段10bへ返すこと（図2のステップS6）、テストスクリプトの削除およびその結果をテスト自動化制御手段10bへ返すこと（図2のステップS7）、テストスクリプト・ファイル情報の検索およびそれがあった場合それを要求元へ返すこと（図2のステップS8）、テストスクリプト管理情報を要求元へ返すこと（図2のステップS9）をそれぞれ行なう。ここで、1または複数のテストスクリプトとは、上述の障害有無判定用テストスクリプトや、障害特定用テストスクリプトである（以下、同様）。

【0026】また、テスト自動化制御手段10bは、テストスクリプト管理手段10aと、テストスクリプト実行手段10cと、各インターフェース手段10f～10iとの間の橋渡しをするもので、テストスクリプトの入力／実行／出力等の制御を行なうものである。具

体的には、この実施例の場合のテスト自動化制御手段10bは、以下に図3を参照して説明するような動作をするものとしてある。すなわち、テストスクリプトの登録か、テストスクリプト・ファイル情報の要求か、テストスクリプトの実行か、デバックコマンドの実行か、テストの終了の各動作の中から検査者の希望するいずれかの動作を選択する(図3のステップS1～S5)。そして、それぞれの動作に応じ、テストスクリプトの登録をテストスクリプト管理手段10aに要求しそよびその結果をテストスクリプト入力手段10fへ返すこと(図3のステップS6)、テストスクリプト・ファイル情報をテストスクリプト管理手段10aに要求しその結果を要求元へ返すこと(図3のステップS7)、テストスクリプト・ファイル情報をテストスクリプト管理手段10aからテストスクリプト実行手段10cへ渡すこと(図3のステップS8)、デバックコマンドをテストスクリプト実行手段10cに渡しそその実行結果を要求元に返すこと(図3のステップS9)をそれぞれ行なう。また、図示は省略しているが、このテスト自動化制御手段10bは、テストスクリプト管理手段10aからテストスクリプトの内容を受け取り、この内容に応じテスト対象のプログラム(実行オブジェクトということもある。)の特定や実行履歴保存環境の設定などをも行なう。

【0027】また、テストスクリプト実行手段10cは、選択されているテストスクリプトの指示に従って必要な情報のダンプ及びログ情報の出力を行なうデバッグの機能を備したものである。具体的には、この実施例の場合のテストスクリプト実行手段10cは、以下に図4を参照して説明するような動作をするものとしてある。テストスクリプトの情報をテストスクリプト管理手段10aからテスト自動化制御手段10bを介して受け取る(図4のステップS1)。また、実行オブジェクトすなわちテスト対象のプログラムをターゲットシステム20(図1参照)にダウンロードする(図4のステップS2)。前記受け取ったテストスクリプトの情報に基づき指定されたアドレスにブレークポイントを設定する(図4のステップS3)。次に、テストモード(障害有無判定モード)のかでデバックモード(障害特定モード)のかを判定する(図4のステップS4)。テストモードの場合はテスト合否判定手段に判定用データを送り、デバックモードの場合はプログラム障害特定手段に特定判定用データを送る(図4のステップS5、S6)。なお、合否判定用のデータや障害特定判定用のデータは対応するテストスクリプト中に記載されている。合否判定手段10dにおいてダンプされたデータを、実行履歴保存環境の設定に従い保存する(図4のステップS7)。実行結果をテスト自動化制御手段に返送する(図4のステップS8)。

【0028】また、テスト合否判定手段10dは、テスト対象のプログラムとのインターフェース部分となるも

のであり、テスト項目と一致しているか否かをチェックするものである。具体的には、この実施例の場合のテスト合否判定手段10dは、以下に図5を参照して説明するような動作をするものとしてある。先ず、合否判定用のデータをテストスクリプト実行手段10cから受け取ると共にテスト対象のプログラムを実行する(図5のステップS1)。テスト対象のプログラムが判定位置(所定ステップ)に来たか否かを判断する(図5のステップS2)。所定ステップに来た場合は必要なデータをダンプしかつその判定を行ないかつ判定結果をテストスクリプト実行手段に返し(図5のステップS3、S4)、そうでない場合はログ情報を出し、テストプログラムを継続実行する(図5のステップS5)。

【0029】また、プログラム障害特定手段10eは、テスト対象のプログラムにおいて障害がどこで発生しているかを特定するものである。ただし、このプログラム障害特定手段10eは、検査者によってデバッグモード(障害特定モード)が選択された場合に動作する。具体的には、この実施例の場合のプログラム障害特定手段10eは、以下に図6を参照して説明するような動作をするものとしてある。先ず、障害特定用のデータをテストスクリプト実行手段10cから受け取ると共にテスト対象のプログラムをステップトレース実行する(図6のステップS1)。テスト対象のプログラムが判定位置(所定ステップ)に来たか否かを判断する(図6のステップS2)。所定ステップに来た場合は必要なデータをダンプしかつ上記障害特定用データを用いその判定を行なう(図6のステップS3)。判定結果が正しくない場合(障害位置である可能性が低い場合)、ログ情報を出力すると共にテスト対象のプログラムを継続実行した後障害特定モードを終了する(図6のステップS5)。判定結果が正しい場合はログ情報を出した後(図6のステップS6)、さらに別の障害特定のステップがないか否かが判定され(図6のステップS7)、ない場合は判定結果をテストスクリプト実行手段10cに返し(図6のステップS8)、ある場合はテスト対象のプログラムのステップトレースの実行を継続する(図6のステップS9)。

【0030】また、テストスクリプト入力手段10fは、どのようなテストを行ないたいかを、検査者30がプログラムテスト装置10に入力するためのものであり、例えばコンピュータシステム40に備わるキーボードやマウスなどを含む手段などで構成出来る。この実施例の場合のテストスクリプト入力手段10fは、以下に図7を参照して説明する様な動作をするものとしてある。テストスクリプトの入力が終了か否かを検査者に問う(図7のステップS1)。入力終了でない場合はテストスクリプトを新規に作成するか否かを検査者に問う(図7のステップS2)。新規作成の場合は、プログラムテスト装置の表示装置(図示せず)に予め定めた新規

テストスクリプト入力用のシートを表示する（図7のステップS3）、一方、新規作成でない場合は、既に作成されたテストスクリプトの変更等と判断してテスト自動化制御手段10bを介しテストスクリプト管理手段10aにテストスクリプトファイルの確認（図7のステップS4）およびテストスクリプト情報を要求し、これによりテストスクリプト管理手段10aからこのテストスクリプト情報を得、この情報に基づき所定の入力シートを上記表示装置に表示する（図7のステップS5）。次に、新規、既存いずれの場合も検査者の入力を待つ（図7のステップS6）。次に、入力完了か否か、必要事項の入力が済んだか否かを判定し（図7のステップS7、S8）、完了の場合は入力データをテスト自動化制御手段10bに送る（図7のステップS9）。この入力データはテストスクリプト管理手段10aのファイルに登録されるが、その際に返送される結果で登録完了か否かを判定する（図7のステップS10）。そして完了した旨の表示（図7のステップS11）若しくはエラー部分の強調表示（例えば反転表示）をする（図7のステップS12）。

【0031】また、テスト結果出力手段10gは、テスト結果を例えはある決められたフォーマット（詳細は後述する）で出力するもので、例えコンピュータシステム40に備わる表示装置やプリンタなどを含む手段で構成出来る。この実施例の場合のテスト結果出力手段10gは、以下に図8を参照して説明する様な動作をするものとしてある。先ず、検査者によるテストスクリプトファイルの指定があるか否かを判定する（図8のステップS1）。ある場合は指定のテストスクリプトファイルに関してのデータを、指定がない場合は既に指定されているテストスクリプトファイルに関してのデータを、テスト自動化制御手段10bに要求する（図8のステップS2、S3）。テスト自動化制御手段10bはテストスクリプト管理手段10aを検索しその結果をテスト結果出力部に返す。この返された結果から、テスト結果出力手段10gはデータの有無を判定し（図8のステップS4）、このテストスクリプトファイルに関するデータがテストスクリプト管理手段になかった場合はエラーメッセージを表示し（図8のステップS5）、あった場合は出力先の確認、出力フォーマットの確認、指定フォーマットによる指定出力先への出力、終了メッセージの表示を順次に行なう（図8のステップS6～S9）。

【0032】また、モード切換／実行手段10hは、テストモード（障害有無判定モード）かデバッグモード（障害特定モード）かのモード切換えを行なって実行するものである。この実施例の場合のモード切換／実行手段10hは、以下に図9を参照して説明する様な動作をするものとしてある。テストモードか否かを問う（図9のステップS1）。そうである場合はプログラムテスト装置10をテストモードすなわち障害有無判定用テ

ストスクリプトを用いる処理モードに（図9のステップS2）、そうでない場合はプログラムテスト装置10をデバッグモードすなわち障害特定用テストスクリプトを用いる処理モードに、する（図9のステップS3）。次に、テスト対象のプログラムを実行するか否かを問い合わせ、検査者の対応に応じた処理をする（図9のステップS4、S5）。

【0033】また、デバッグコマンド入力手段10iは、コンピュータシステム40の一部で構成してあるデバッガに具備されているデバッガの機能を利用可能とするコマンドを入力するものである。この実施例の場合のデバッグコマンド入力手段10iは、以下に図10を参照して説明する様な動作をするものとしてある。デバッグコマンド入力画面をプログラムテスト装置10の表示装置（図示せざ）に表示する（図10のステップS1）。検査者によるデバッグコマンドの入力を待ち、また、その入力が終了か否かを問う（図10のステップS2、S3）。入力されたデバッグコマンドをテスト自動化制御手段10bに送る（図10のステップS4）。テスト自動化制御手段10bから返されるデバッグコマンド実行結果を表示装置の画面に表示する（図10のステップS5）。

【0034】2. プログラムテスト動作の説明

2-1. 障害有無判定動作の説明

先ず、テスト対象のプログラムに障害があるか否かを判定する方法（第一発明のプログラムテスト方法）の実施例について説明する。

【0035】検査者は障害有無判定用テストスクリプトを作成するためにテスト実施内容を上記のテストスクリプト入力手段10fを利用して入力する。この障害有無判定用テストスクリプトの内容はテスト対象のプログラムの仕様などを考慮して予め検査者により検討されるものである。理解を深めるため、障害有無判定用テストスクリプトの一例を、図11（A）に示した。この図11（A）に示した例は、テスト対象のプログラム（以下、テスト対象プログラムAともいいう。）を含むファイルの名51が、test.Aであり、また、実行履歴保存ファイル53（後述する）が、test.A.log1というファイル名とされ、所定のブレークポイント（所定のステップ）55が、番地××××であり、ダンプするべきデータ57が変数B、C、DおよびアドレスA1、A2であり、所定のテスト項目59が、Bが10であるか否かのテストである例を示している。なお、障害有無判定用テストスクリプトの入力においては、図7を用いて既に説明した様に、テスト実施内容の記述チェックを行ない不備があつた場合は修正が要求される。不備がない場合は、この障害有無判定用テストスクリプトはテストスクリプト管理手段10aに登録される。

【0036】また、検査者は上記のモード切換／実行手段10hを利用してテストモードを選択する。する

と、テスト自動化制御手段10aは、テストスクリプト管理手段10aに対して障害有無判定用テストスクリプトの情報を要求する。これに応じテストスクリプト管理手段10aは、テスト対象プログラムA用の障害有無判定用テストスクリプトこの場合は上記のファイルtest Aにある障害有無判定用テストスクリプトの情報を自動化制御手段10cに送る。

【0037】すると、テスト自動化制御手段10bは、この障害有無判定用テストスクリプトからテスト対象プログラムを構成しているファイル名この場合はtest Aを識別する。次に、実行履歴保存用のファイル名この場合はtest A.logを識別しそのためのファイルをオープンすることで実行履歴環境を設定する。その後、テスト対象プログラムAとその他のスクリプト情報(図11(A)の5.3～5.9で示されるもの)とをテストスクリプト実行手段10cに送る。テストスクリプト実行手段10cではロードファイル名test Aにおける実行オブジェクト(テスト対象プログラムA)をターゲットシステム20にダウンロードすること、障害有無判定用テストスクリプトで指定されたアドレス(ここでは図11(A)の5.5参照)にブレークポイントを設定すること、障害有無判定用テストスクリプトで指定されたダンプすべきデータ(ここでは図11(A)の5.7～5.9参照)およびテスト項目(ここでは図11(A)の5.9参照)を合否判定手段10dに送ること、テスト対象プログラムAの実行をそれぞれ行なう。テスト合否判定手段10dでは設定したブレークポイントでテスト対象プログラムAが停止した時に受け取ったテストスクリプトのダンプデータの指示に従ってデータのダンプを行なうと共にテスト項目に示されたテストを行なう。その後ダンプしたデータとテスト合否判定結果をテストスクリプト実行手段10cに送る。テストスクリプト実行手段10cは、この合否判定結果をテスト結果出力手段10gおよびテストスクリプト管理手段10aに送る。テスト結果出力手段は10gは、図8を用いて説明した様に、予め決められたフォーマットで合否判定結果を出力する。また、テストスクリプト管理手段10aは、この合否判定結果をテスト対象プログラムA用の障害有無判定用テストスクリプトの管理情報として記録する。以上で、テスト対象プログラムAの障害有無判定が行なえる。他にもテスト対象のプログラムがある場合は上述の処理を繰り返して実施する。

【0038】この上述の障害有無判定方法では、障害有無判定用テストスクリプトに所望のテスト内容を記入することで、人手を介すことなく自動的にテスト対象のプログラムを実行しこのプログラム中の所定ポイント(所定ステップ)で、データのダンプおよびテスト合否判定を行なってこのプログラム中の障害の有無の判定結果を出力する。このため、大量的テスト対象のプログラムについて障害の有無を効率的にテストできる。

【0039】2-2. 障害特定動作の説明

上記2-1項においては、大量のテスト対象のプログラムそれぞれについてそれらが障害を持つものか否かのテストを人手を介すことなく自動的かつ効率的に行なう方法を示したが、プログラムの開発工程の次の段階では障害を持つプログラムのどこに障害の原因があるのか調べる必要(障害を特定する必要)がある。これを行なう方法(第二発明の方法)について以下説明する。

【0040】検査者は、障害を含んでいたためテストモードにおいて不合格とされたプログラムに対応する障害有無判定用テストスクリプトを、テストスクリプト入力手段10fを利用して、プログラムテスト装置10の表示装置(図示せず)に表示させる。次に、この障害を含むプログラムの障害箇所特定に好適なテストスクリプトすなわち障害特定用テストスクリプトを、テストスクリプト入力手段10fを利用して、テストスクリプト管理手段10aの記憶部のファイルに登録する。この登録する障害特定用テストスクリプトは、従来技術において説明した障害解析関数とは次の様な点で大きく相違する。すなわち、第一発明の方法で障害があると判定された際に得られたテスト結果から見て当該障害の特定に好適なように、障害有無判定用テストスクリプトを変更して(流用して)作成される点で大きく相違する。図11(B)に、図11(A)に示した障害有無判定用テストスクリプトを流用して作成した障害特定用テストスクリプトの例を示した。この図11(B)に示した例は、障害有無判定のためのテストで得た結果から見て、ロードファイル61、実行履歴ファイル63の内容は障害有無判定用テストスクリプトの場合のまで良い例であり、ただし、ブレークポイント65はy.y.yとし、ダンプすべきデータ67は変数AおよびアドレスA1、A2とし、テスト項目69はAが1.4であるか否かのテストとそれぞれ変更するのが、障害特定に好適である例である。なお、障害特定用テストスクリプトの入力、修正、登録などは、障害有無判定用テストスクリプトの入力、修正、登録手順と同様にテストスクリプト入力手段を利用して行なわれるので、その説明は省略する。

【0041】次に、検査者はモード切換え／実行手段10hを利用してプログラムテスト装置10のモードをデバッグモードにする。すると、テスト自動化制御手段10bは、実行オブジェクトであるロードファイル名とモードの識別および実行履歴保存環境の設定を障害有無判定時と同様な手順で行ない、また、このロードファイル名をその他の障害特定用テストスクリプト情報(図11(B)の6.5～6.9)と共にテストスクリプト実行手段10cに送る。テストスクリプト実行手段10cでは実行オブジェクト(テスト対象プログラムA)をターゲットシステムにダウンロードすること、障害特定用テストスクリプトで指定されたアドレス(ここでは図11(B)の6.5参照)にブレークポイントを設定すること、障害特定用テストスクリプトで指定されたダンプす

べきデータ（ここでは図11（B）の67参照）およびテスト項目（ここでは図11（B）の69参照）を合否判定手段10dに送ること、プログラム障害特定手段10eにテスト対象プログラムAの実行制御を依頼することを、それぞれ行なう。プログラム障害特定手段10eは、テスト対象プログラムAをステップトレース実行しながら、障害特定用テストスクリプトで指示されたステップでのデータのダンプやテスト項目を実施する。障害が検知された場合、その旨はテストスクリプト実行手段10c経由でテスト自動化制御手段10bに通知される。テスト自動化制御手段10bはこの通知された障害特定のための情報をプログラムテスト装置10の表示装置やプリンタ（図示せず）等の出力装置上に出力する。検査者は出力された情報における障害発生位置情報や、プログラム障害特定手段10eで収集された実行履歴情報を利用し、障害原因の解析を行ない障害を特定する。なお、さらに解析に必要な情報を調べたい場合は、通常デバッガで提供されているデバッグコマンドがこのデバッグモードでは利用可能であるので、このデバッグコマンドをデバックコマンド入力手段10iを利用して入力することで、必要な情報を得ることが出来る。

【0042】この第二発明に係る障害特定の方法によれば、障害解析の箇所を変更するごとに障害解析関数を変更しこの変更した関数をデバッグ制御プログラムに組み込むという手間が必要なく、障害有無判定のテスト過程で使用した障害有無判定用テストスクリプトを単に変更して得た障害特定用テストスクリプトを用い障害箇所の特定が可能となる。また、検査者の判断を求めるながら、障害発生箇所を特定する必要もなくなるので、障害解析の所要時間が軽減される。

【0043】

【発明の効果】上述した説明から明らかなように、第一発明のプログラムテスト方法によれば、テスト対象のプログラムに応じた障害有無判定用テストスクリプトを登録する。そして、テスト対象のプログラムをテスト実行すると共に障害有無判定用テストスクリプトに基づくチェックを自動的に行ないその結果を出力できる。このため、大量のテストプログラムの実行とこれらプログラムにおける障害の有無の判定とを、人手を介さずに自動的に実行できるので、プログラムのテストを従来に比べ効率的にかつ人為的なミスの少ない条件で行なえる。

【0044】また、第二発明のプログラムテスト方法によれば、障害特定に好適な障害特定用テストスクリプトを登録する。そして、テスト対象のプログラムをテスト

実行すると共に障害特定用テストスクリプトに基づくチェックを自動的に行ないその結果を出力できる。このため、テスト対象のプログラムに対する障害有無の判定および障害特定という一連の処理を自動的に行なえる。また、ここで用いる障害特定用テストスクリプトは、障害有無判定用テストスクリプトを用いたテストの結果に基づいて作成するものであるので、熟練をそれほど必要とすることなく、作成できる。したがって、デバッグ制御プログラムに障害解析関数を組み込む従来技術で問題となっていた、高いスキルが必要という問題と時間がかかるという問題とを解決出来る。

【0045】また、この出願の第三発明のプログラムテスト装置によれば、第一発明の実施を容易とする。また、この出願の第四発明のプログラムテスト装置によれば、第二発明の実施を容易とする。

【図面の簡単な説明】

【図1】実施例のプログラムテスト装置の説明図である。

【図2】テストスクリプト管理手段の説明図である。

【図3】テスト自動化制御手段の説明図である。

【図4】テストスクリプト実行手段の説明図である。

【図5】テスト合否判定手段の説明図である。

【図6】プログラム障害特定手段の説明図である。

【図7】テストスクリプト入力手段の説明図である。

【図8】テスト結果出力手段の説明図である。

【図9】モード切換え／実行手段の説明図である。

【図10】デバックコマンド入力手段の説明図である。

【図11】テストスクリプトの説明図であり、（A）は障害有無判定用テストスクリプトの説明図、（B）は障害特定用テストスクリプトの説明図である。

【符号の説明】

10：実施例のプログラムテスト装置

10a：テストスクリプト管理手段

10b：テスト自動化制御手段

10c：テストスクリプト実行手段

10d：テスト合否判定手段

10e：プログラム障害特定手段

10f：テストスクリプト入力手段

10g：テスト結果出力手段

10h：モード切換え／実行手段

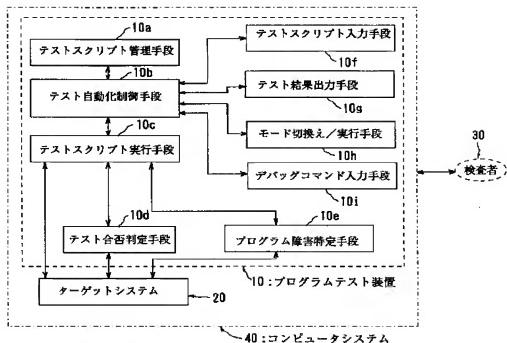
10i：デバックコマンド入力手段

20：ターデットシステム

30：検査者

40：コンピュータシステム

【図1】



実施例のプログラムテスト装置の説明図

(A)

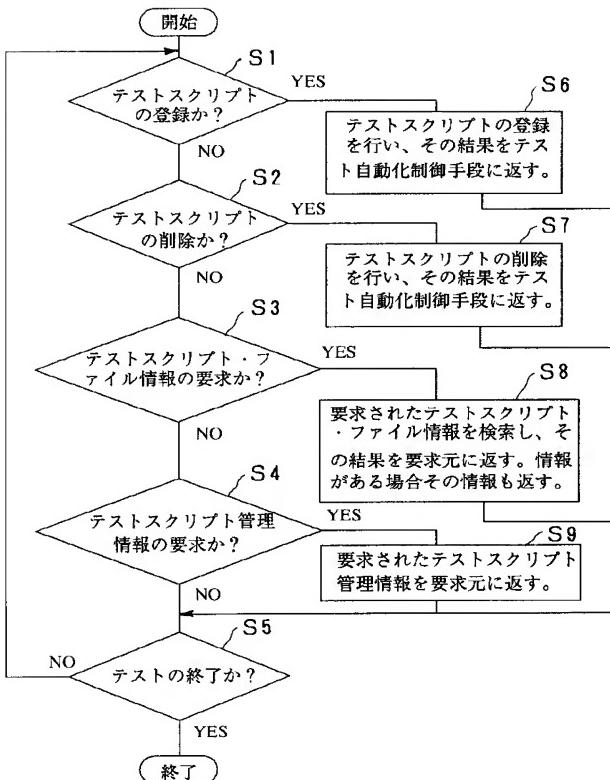
【図11】

ロードファイル:	testA	→ 51
実行履歴ファイル:	testA.log	→ 53
ブレークポイント:	BP XXXX	→ 55
ダンプデータ:	変数: B, C, D アドレス: A1, A2	→ 57
テスト項目:	? B = 10	→ 59

ロードファイル:	testA	→ 61
実行履歴ファイル:	testA.log	→ 63
ブレークポイント:	BP yyyy	→ 65
ダンプデータ:	変数: A アドレス: A1, A2	→ 67
チェック項目:	7 A = 14	→ 69

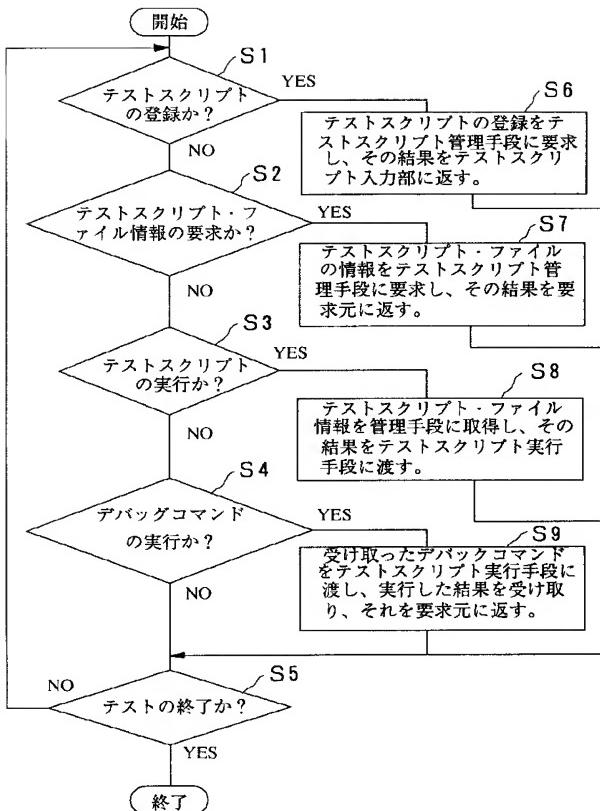
テストスクリプトの説明図

【図2】



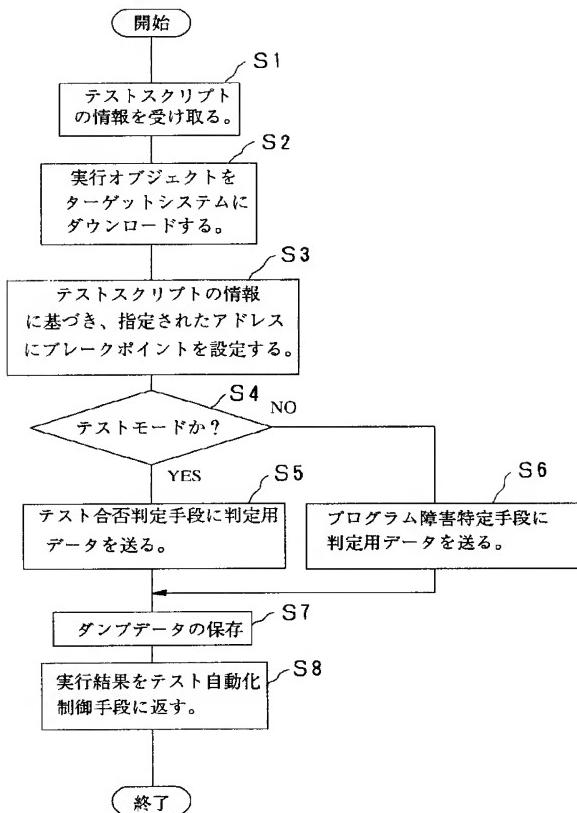
テストスクリプト管理手段の説明図

【図3】



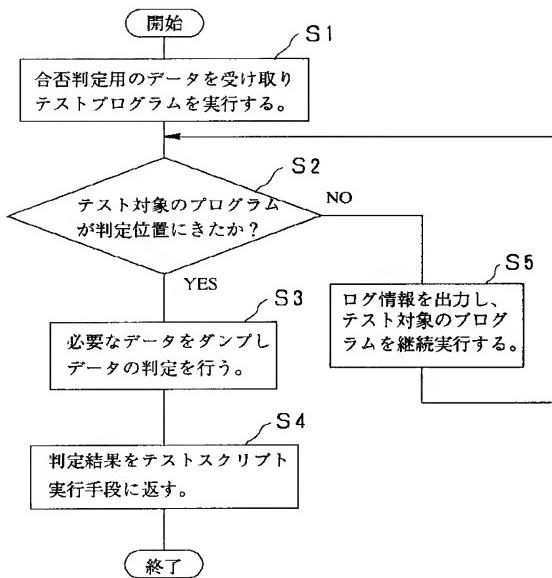
テスト自動化制御手段の説明図

【図4】



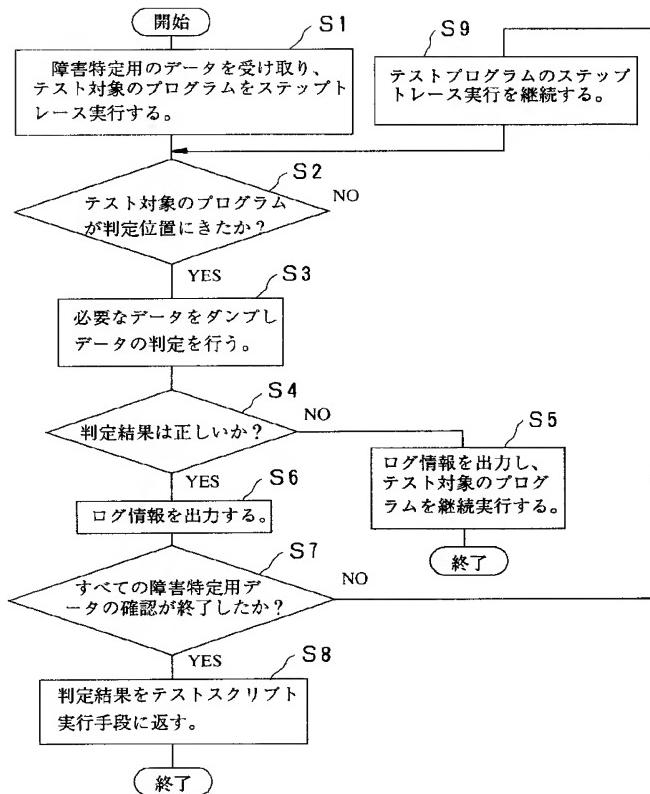
テストスクリプト実行手段の説明図

【図5】



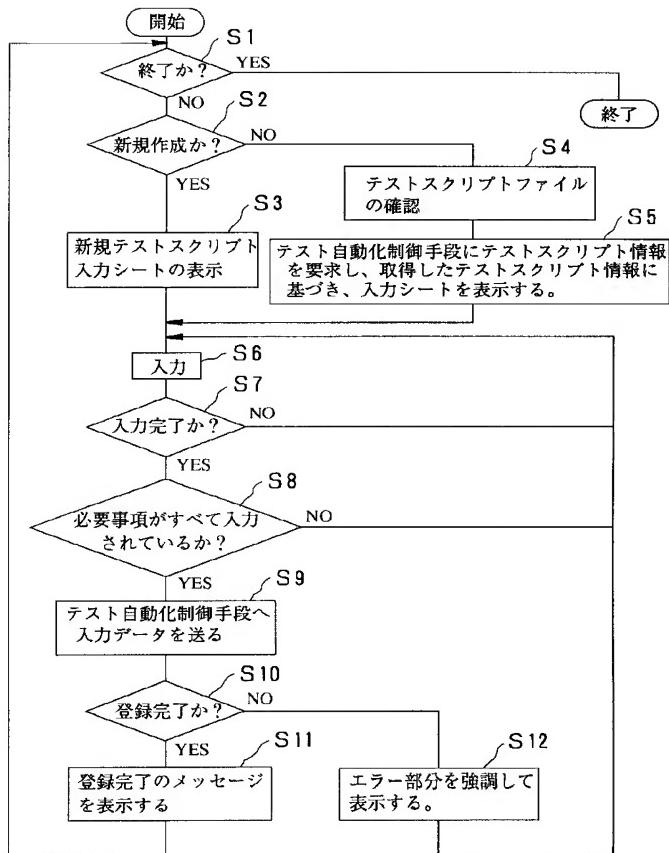
テスト合否判定手段の説明図

【図6】



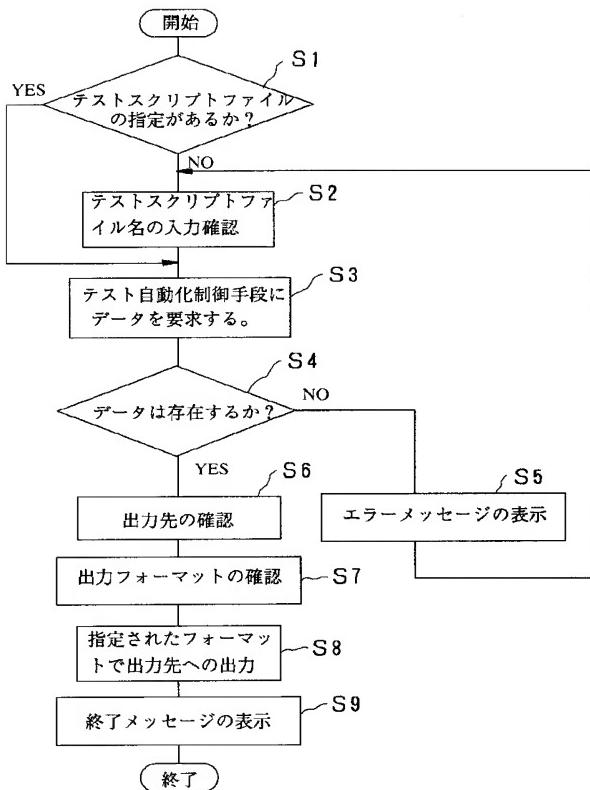
プログラム障害特定手段の説明図

【図7】



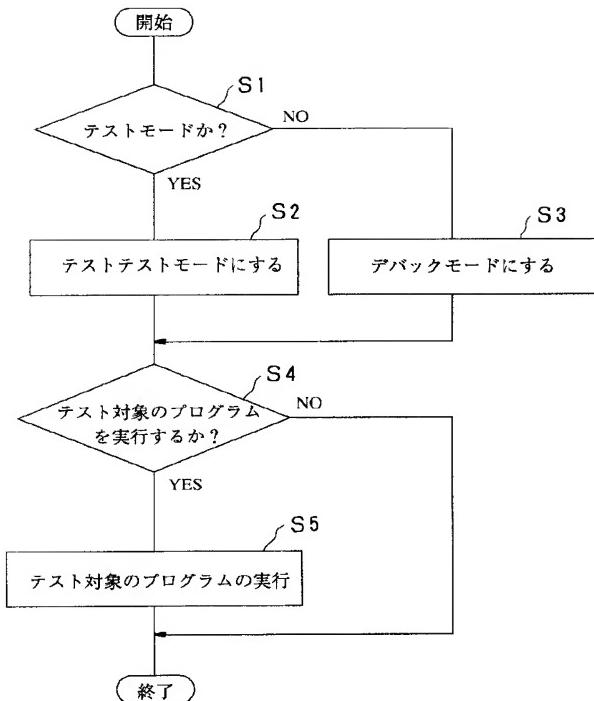
テストスクリプト入力手段の説明図

【図8】



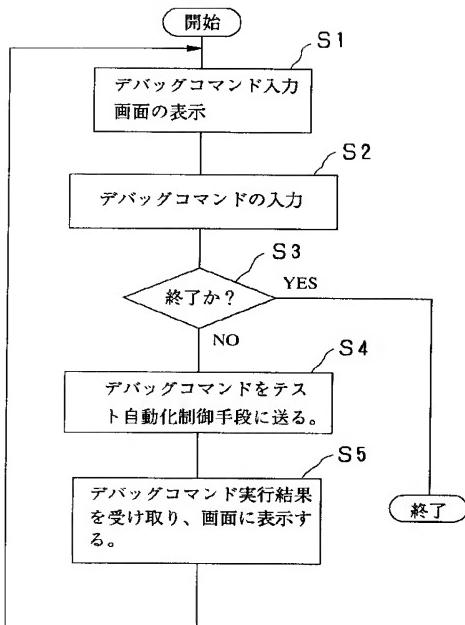
テスト結果出力手段の説明図

【図9】



モード切換え／実行手段の説明図

【図10】



デバッグコマンド入力手段の説明図